Cell Projection of Convex Polyhedra

Stefan Roettger and Thomas Ertl

Visualization and Interactive Systems Group [†]
University of Stuttgart

Abstract

Finite element methods commonly use unstructured grids as the computational domain. As a matter of fact, the volume visualization of these unstructured grids is a time consuming task. Here, the fastest known object order algorithm is the projected tetrahedra algorithm of Shirley and Tuchman. Even with the upcoming of programmable graphics hardware, the rendering performance did not keep up with the growing complexity of the simulation data. In this paper we strive to improve the performance of the cell projection technique by posing several restrictions on the optical model. This allows us to devise a simple but fast hardware-accelerated algorithm which is able to project arbitrary polyhedral cells, that is tetrahedra, prisms, hexahedra, etc. For this reason, our algorithm is well suited for the display of unstructured FEM meshes with mixed cell types, but it is also applicable to the real-time display of gaseous phenonema, such as fire and ground fog.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling, I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism.

Keywords: Direct volume rendering, unstructured grids, cell projection.

1. Introduction

In the area of volume visualization the availability of programmable graphics hardware has lead to both improved performance and rendering quality. In the case of regular data the pre-integration technique[3] is the predominant recent improvement. While the pre-integration technique has been applied to unstructured tetrahedral grids even before[12], the performance of unstructured volume rendering methods is still poor compared to the regular case. In this paper we try to narrow the performance gap by posing several restrictions on the optical model. This allows us to devise an algorithm which efficiently utilizes the graphics hardware to speed up unstructured volume rendering.

Typically, unstructured grids are generated by finite element methods. In order to visualize the data, all cells first

have to be sorted in a back to front fashion[18, 1]. After that, each cell is decomposed into tetrahedra which can be displayed efficiently using the well known Projected Tetrahedra (PT) algorithm of Shirley and Tuchman[14, 15]. Actual implementations of this algorithm achieve a peak performance of 250,000[19] to 600,000[4] tetrahedra per second including times for sorting. Due to the growing complexity of the simulation data frame rates of less than one frame per second are still quite common for typical unstructured data sets.

Recently, hardware-accelerated methods have been proposed to speed up the PT algorithm, but with actual graphics hardware still no more than approximately 480,000[16] to 490,000[20] tetrahedra are possible (timings do not include sorting). There also exist hardware concepts to overcome the speed limitations, but it is uncertain when these concepts will find its way into graphics accelerators[6]. Since recent efforts to significantly speed up the PT algorithm have not led to satisfactory results, we pursue a different strategy in this paper: First we evaluate the theoretical limit on the number of polyhedra that can be rendered on actual graphics hardware. Based on these results we propose a reasonable modification of the optical model to approach the theoretical limit.

2. Theoretical Performance

In principal, all the faces of an unstructured data set have to be treated to reconstruct the ray integral exactly. For the

---

[†] University of Stuttgart, Computer Science Institute, VIS Group, Breitwiesenstrasse 20-22, 70565 Stuttgart, Germany; E-mail: Stefan.Roettger@informatik.uni-stuttgart.de.

case of hexahedral cells, this results in 6 faces with 4 vertices each. Assuming that the volumetric grid can be rendered with triangle stripping, 8 vertices have to be passed down the graphics pipeline per hexahedron. Actual graphics accelerators like the NVIDIA GeForce3 reach a peak performance of about 12 million vertices per second using triangle strips (in practical experience). Thus, the maximum theoretical performance of the NVIDIA GeForce3 is 1.5 million hexahedra per second.

In order to verify the theoretical result, we first applied maximum intensity projection (MIP)[5]. The advantage of MIP is that a volumetric grid can be visualized just by rendering all the faces of the cells in an unsorted order. Without great loss of accuracy the scalar values can be assumed to vary linearly inside each hexahedron. Then the maximum projected scalar value of each ray segment is either the value on the front or on the back face. Using this approach we achieved a performance of 643,000 hexahedra or 5.1 million triangles per second. Assuming that a hexahedron needs to be decomposed into at least 5 tetrahedra to be rendered with the PT algorithm the experimental result of 643,000 hexahedra per second corresponds to 3.2 million tetrahedra per second. This is still far away from the theoretical maximum, but it is almost a magnitude faster than the best known PT implementation.

The performance for such a simple optical model like MIP is already considerably lower than the theoretical limit. This is mainly due to the large rasterization overhead. Hence, it is no surprise that the performance is even worse in the case of the standard volume density optical model[17]. This is due to the requirement of visibility sorting. Conceptually, the tetrahedra must be read, written, and read back from main memory for sorting (compare Wittenbrink et. al[19]). With increasing rendering speed of the graphics accelerator the memory bandwidth consumed by visibility sorting becomes the limiting factor. This behaviour starts at approximately 1.5 million tetrahedra per second on actual PC hardware. Since the total performance is currently only around 600,000 tetrahedra per second the main limiting factor is still the graphics accelerator (and the CPU). We suspect that a significant performance bump beyond the mentioned 1.5 million tetrahedra per second limit is possible only with a structural paradigm shift of graphics accelerators or special purpose hardware.

3. Projected Polyhedra Algorithm

Because of the limiting behaviour of visibility sorting, we devise an efficient algorithm for an emissive optical model[8] which does not require sorting. In our opinion this optical model can be considered to be a good tradeoff between speed and quality. The emissive optical model neglects absorption so that the ray integral is simply the sum of all emissions along each ray. As a welcome side effect sorting is not required, since the blend function is commutative. In comparison to the standard optical model the emissive model gives less visual clues but as we will see the implementation is extremely simple so that it can serve as a fast preview and prototyping option.

Recently, Mech[9] proposed a method to render bounded layered fog using an emissive optical model. The bounded fog is defined within a triangular surface mesh which allows for easy hardware-accelerated computation of the ray integral. The length of each ray segment is calculated in the frame buffer by coding the distance from the near clipping plane into the vertex color. Then the length of the ray segments can be computed by rendering the back faces of the fog boundary and by subtracting the front faces. While this approach is simple yet very fast, it assumes a constant fog density and requires a 12 bit visual to eliminate Mach bands. In the following we extend this algorithm to project arbitrary cell types, such as tetrahedra, hexahedra, or prisms, without the restriction to a 12 bit frame buffer and with linearly interpolated densities within each cell.

Our so called Projected Convex Polyhedra (PCP) algorithm requires three passes per cell. In the first two passes the normalized length of the ray segments is calculated in the alpha channel of the frame buffer. For this purpose, the distance d to the near plane is computed for each vertex of the cell. Let d_{max} denote the maximum distance per cell, let d_{min} denote the minimum distance, and let $\Delta d = d_{max} - d_{min}$ be the difference of both (see also Figure 1). Then the back faces of a cell are rendered into the alpha channel of the frame buffer with the alpha component of each vertex set to $\alpha = (d - d_{min})/\Delta d$. In the same fashion, the front faces of the cell are rendered into the alpha channel with subtractive blending enabled. As a result, the normalized ray segment lengths are now available in the alpha channel of the frame buffer.
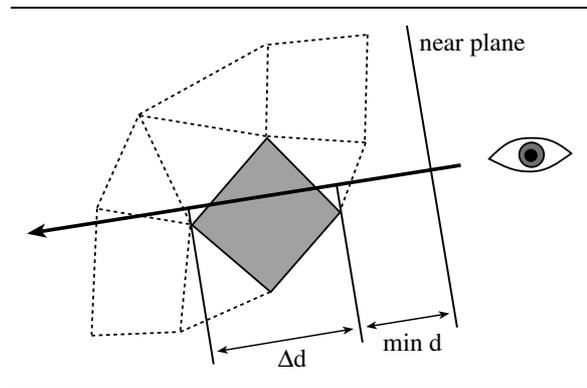

Figure 1: *Projection of polyhedral cells.*

In the last pass all faces of the cell are rendered into the color channel of the frame buffer. Let $\kappa(S)$ denote the transfer function of the emissive optical model depending on the scalar value S. Then the color I

of each vertex is set to $I = \kappa(S)\Delta d/2$ using the following blend function in OpenGL notation: `glBlend-Func(GL_DST_ALPHA,GL_ONE)`. This effectively multiplies the average emission along each ray segment with the segment length already stored in the alpha channel.

In contrast to Mech's method we do not require a 12 bit visual, since we use normalized ray segment lengths for each cell. Another solution to suppress the Mach bands would be to use the floating point render target of actual PC graphics accelerators such as the ATI Radeon 9700. However, since the algorithm is mainly rasterization bound the increased bandwidth for the floating point render target would significantly slow down rendering. Also Mech's algorithm is not as flexible as ours. Using our method, almost any desired volumetric object or (emissive) effect can be constructed from tetrahedra, prisms, and hexahedra in a very compact way (compare Figure 2).
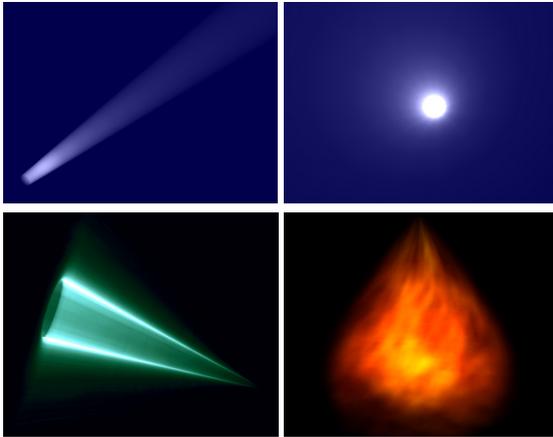

Figure 2: *Synthetic data sets: A search light with quadratic intensity attenuation (top row), a laser cone (bottom left), and a campfire generated with 3D Perlin noise (bottom right).*

4. Results

In principle, all types of cells used for FEM such as tetrahedra, hexahedra, prisms, pyramids etc. are compatible with our approach. For the common case of projected hexahedra Schussman et al.[13] report about 80,000 hexahedra per second. We achieve about 212,000 hexahedra per second, which is a performance increase of 265%. Compared to the 643,000 hexahedra per second of the MIP method, the performance difference is mainly due to the increased number of rendering passes (3 instead of 1).

In Figure 3 and 2 example data sets are shown that have been visualized with the PCP algorithm. The corresponding timings are given in Table 4. To speed up projection hexahedra with zero emission were discarded.
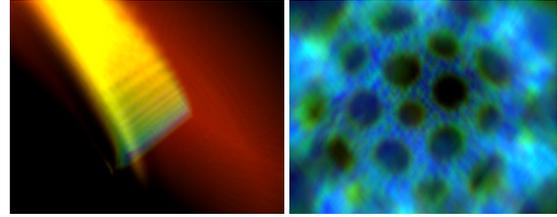
Figure 3: *Blunt Fin and Bucky Ball data set.*

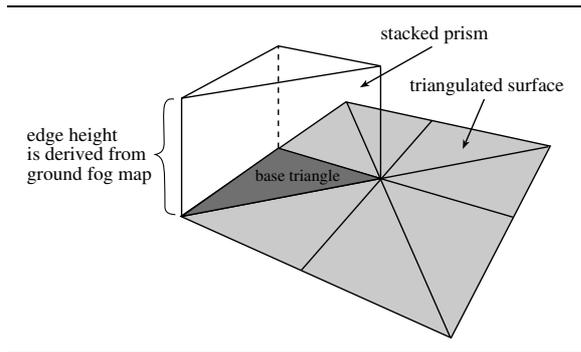| Data set	dimension	#hexahedra	frames per sec.
BluntFin	$32{\times}32{\times}40$	37,479	8.5
Bucky Ball	$32{\times}32{\times}32$	29,791	15.9
Search Light	$16{\times}4{\times}32$	1,395	115.8
Camp Fire	$16{\times}16{\times}16$	3375	51.3

Figure 4: *Timings for hexahedral projection.*

5. Application Example: Ground Fog

Besides the application area of scientific volume visualization as demonstrated in Figure 3 the performance and flexibility of the proposed cell projection algorithm paves the way for other fields of application. As an example, we demonstrate the real-time display of natural gaseous phenomena. In principle, all effects related to light-emitting gas can be modeled. In particular, the display of ground fog in terrain rendering scenarios benefits from our algorithm, as shown in the following.

In a terrain rendering scenario the landscape is commonly given as a height field. Here, the basic idea to display ground fog is to use a second height field (the ground fog map) which defines the height of the fog layer above the ground. Each triangle of the surface mesh is treated as a base triangle onto which a vertically aligned prism is stacked. The height of the prisms, that is the heights of the three vertical edges of each prism, are derived from the ground fog map (see Figure 5).

At the top left of Figure 6 an example height field of Yukon Territory, Canada, is depicted. The shown ground fog has been generated with 2D Perlin noise[10]. In order to reduce the number of displayed triangles and stacked prisms, we used a *continuous level of detail* (C-LOD) approach[7, 2]. The applied terrain rendering algorithm[11] also implements geomorphing so that the popping effect is suppressed efficiently. This allows the viewer to fly through the ground fog without experiencing any temporal aliasing artifacts. We achieved an average frame rate of approximately 25 Hertz for a window size of $512{\times}384$. Inside the fog, we have to take care of prisms that intersect the near clipping plane.

Figure 5: *Stacking prisms onto a triangulated surface.*

Figure 6: *Ground fog generated with 2D Perlin noise.* ***Top left***: *Emissive optical model.* ***Top right***: *Maximum Intensity Projection (MIP).* ***Bottom***: *Aurea Borealis (polar light).*

In such a case the ray segment lengths are partially invalid, since the corresponding back faces are not rendered completely. To circumvent this problem, we also render the intersection of each prism with the near clipping plane with $\alpha = (d_{near} - d_{min})/\Delta d$ after the second pass. The same strategy is necessary after the third pass: The intersection of each prism with the near clipping plane is rendered with correctly interpolated colors to avoid the partial display of clipped tetrahedra.
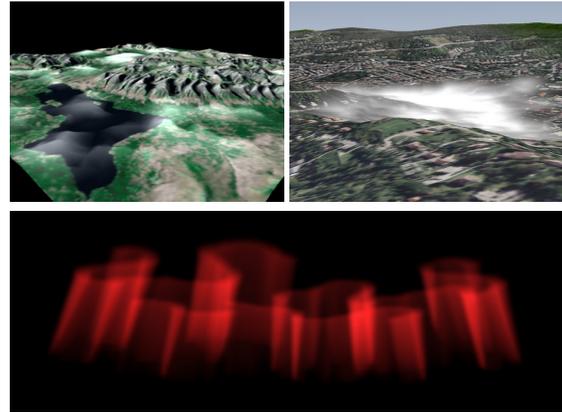
The ground fog in the valley as shown at the top right of Figure 6 is displayed with maximum intensity projection. The corresponding height field has been painted by hand with a standard image manipulation application. Since the MIP method requires only one pass in comparison to the three passes of the PCP algorithm the rasterization bottleneck is reduced significantly. This leads to more than twice the frame rate (> 50 Hz) as in the previous example.

Despite the seemingly unsuitable optical model we have found a reasonable setup for the MIP method: The fog's optical density is set to zero at the bottom of the prisms. At the top of the prisms the density correlates to the height of the fog layer. Although this setting does not reproduce the fog physically it is well suited for the real-time display of foggy areas in interactive entertainment where fog can be used as a game play relevant element.

Another application area of the described ground fog rendering method is the display of the Aurea Borealis, since polar light is a purely emissive natural phenomenon. We set up a height field that corresponds to the penetration depth of the particles into the earth's ionosphere. The result of this procedure can be seen at the bottom of Figure 6.

6. Conclusion

We have presented two fast yet simple cell projection algorithms ,namely the MIP and the PCP method, that are suited for maximum intensity projection or an emissive optical model, respectively. The algorithms are capable of projecting arbitrary polyhedral cells. Therefore they can be applied to visualize mixed type meshes that are generated by FEM simulations. Due to their flexibility, they are also suited for the display of volumetric effects in interactive entertainment. We have demonstrated this by rendering fire and ground fog in real-time.

7. Acknowledgements

References

1. J. Comba, J. T. Klosowski, N. L. Max, J. S. B. Mitchell, C. T. Silva, and P. L. Williams. Fast Polyhedral Cell Sorting for Interactive Rendering of Unstructured Grids. *Computer Graphics Forum (Proc. Eurographics '99)*, 18(3):369–376, 1999.

2. M. Duchaineau, M. Wolinsky, D. E. Sigeti, M. C. Miller, Ch. Aldrich, and M. B. Mineev-Weinstein. ROAMing Terrain: Real-Time Optimally Adapting Meshes. In *Proc. Visualization '97*, pages 81–88. IEEE, 1997.

3. K. Engel, M. Kraus, and Th. Ertl. High-Quality Pre-Integrated Volume Rendering Using Hardware-Accelerated Pixel Shading. In *Eurographics Workshop on Graphics Hardware '01*, pages 9–16. ACM SIGGRAPH, 2001.

4. S. Guthe, S. Roettger, A. Schieber, W. Strasser, and Th. Ertl. High-Quality Unstructured Volume Rendering on the PC Platform. In *Proc. EG/SIGGRAPH Graphics Hardware Workshop '02*, pages 119–125, 2002.

5. W. Heidrich, M. McCool, and J. Stevens. Interactive Maximum Projection Volume Rendering. In *Proc. Visualization '95*, pages 11–18. IEEE, 1995.

6. D. King and C. M. Wittenbrink. An Architecture for Interactive Tetrahedral Volume Rendering. In *Proc. IEEE/EG Workshop on Volume Graphics '01*, pages 163–180. Springer Verlag, Wien, 2001.

7. P. Lindstrom, D. Koller, W. Ribarsky, L. F. Hodges, N. Faust, and G. Turner. Real-Time, Continuous Level of Detail Rendering of Height Fields. In *Proc. SIGGRAPH '96*, pages 109–118. ACM, 1996.

8. Nelson L. Max. Optical Models for Direct Volume Rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, 1995.

9. Radomir Mech. Hardware-Accelerated Real-Time Rendering of Gaseous Phenomena. *Journal of Graphics Tools*, 6(3):1–16, 2001.

10. Ken Perlin. An Image Synthesizer. *Computer Graphics (Proc. SIGGRAPH '85)*, 19(3):287–296, 1985.

11. S. Roettger, W. Heidrich, Ph. Slusallek, and H.-P. Seidel. Real-Time Generation of Continuous Levels of Detail for Height Fields. In *Proc. WSCG '98*, pages 315–322. EG/IFIP, 1998.

12. S. Roettger, M. Kraus, and Th. Ertl. Hardware-Accelerated Volume and Isosurface Rendering Based on Cell-Projection. In *Proc. Visualization '00*, pages 109–116. IEEE, 2000.

13. G. Schussman and N. L. Max. Hierarchical Perspective Volume Rendering Using Triangle Fans. In *Proc. TCVG Eurographics Workshop (VolumeGraphics '01)*, pages 309–320. IEEE/EG, 2001.

14. P. Shirley and A. Tuchman. A Polygonal Approximation to Direct Scalar Volume Rendering. *ACM Computer Graphics (San Diego Workshop on Volume Visualization)*, 24(5):63–70, 1990.

15. C. M. Stein, B. G. Becker, and N. L. Max. Sorting and Hardware Assisted Rendering for Volume Visualization. In *Symposium on Volume Visualization '94*, pages 83–89. IEEE, 1994.

16. M. Weiler and Th. Ertl. Hardware-Based View-Independent Cell Projection. In *Proc. IEEE Symposium on Volume Visualization '02*, pages 13–22, 2002.

17. P. L. Williams and N. L. Max. A Volume Density Optical Model. In *Computer Graphics (Workshop on Volume Visualization '92)*, pages 61–68. ACM, 1992.

18. Peter L. Williams. Visibility Ordering Meshed Polyhedra. *ACM Transactions on Graphics*, 11(2):103–126, 1992.

19. C. M. Wittenbrink. CellFast: Interactive Unstructured Volume Rendering. In *IEEE Visualization '99 Late Breaking Hot Topics*, pages 21–24, 1999.

20. B. Wylie, K. Moreland, L. A. Fisk, and P. Crossno. Tetrahedral Projection using Vertex Shaders. In *Proc. IEEE Symposium on Volume Visualization '02*, pages 7–12. ACM Press, 2002.